8/5/1 (Item 1 from file: 351)
DIALOG(R)File 351:Derwent WPI

010705270 **Image available**
WPI Acc No: 1996-202225/199621
XRPX Acc No: N96-169674
Software control system for use in distributed computing environment - has central processor and secondary platforms facilitating launch of software applications within distributed environment
Patent Assignee: BULL SA (SELA)
Inventor: AYDIN A; KADIMA H
Number of Countries: 006 Number of Patents: 002
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
EP 708402	A1	19960424	EP 94402197	A	19941003	199621	B
JP 8185378	A	19960716	JP 95196738	A	19950801	199638	

Priority Applications (No Type Date): EP 94402197 A 19941003
Cited Patents: 05Jnl.Ref; US 4589068
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
EP 708402	A1	F	8	G06F-011/00	
Designated States (Regional): DE ES FR GB IT					
JP 8185378	A		6	G06F-015/16	

Abstract (Basic): EP 708402 A
The system assists in the launch of distributed applications (AC, AS1-AS2) in a distributed computing environment (ENV). This environment includes a central platform (MFC) associated with a number of secondary platforms (HC1, HC2).
It includes a process controller (CNT) for surveillance of the global state of the distributed system embedded in the central platform. There are process agents (AG11-AG23) embedded in the secondary platforms and an assembly of process DAEMONS which serve as interface between the process agents and the process controller. The DAEMONS are dormant processor which can be reactivated by a specific event.
ADVANTAGE - Provides efficient help for DCE application development. Takes into account inherent characteristics of system. Compatible with user friendly interface e.g windows
Dwg.1/8
Title Terms: SOFTWARE; CONTROL; SYSTEM; DISTRIBUTE; COMPUTATION; ENVIRONMENT; CENTRAL; PROCESSOR; SECONDARY; PLATFORM; FACILITATE; LAUNCH; SOFTWARE; APPLY; DISTRIBUTE; ENVIRONMENT
Derwent Class: T01
International Patent Class (Main): G06F-011/00; G06F-015/16
International Patent Class (Additional): G06F-009/06; G06F-011/28
File Segment: EPI

8/5/2 (Item 1 from file: 347)
DIALOG(R)File 347:JAPIO

05229878
DEBUGGING SUPPORT TOOL OF APPLICATION DISPERSED AT INSIDE OFDCE-TYPE DISTRIBUTED INFORMATION PROCESSING ENVIRONMENT

PUB. NO.: 08-185378 **JP 8185378** A]
PUBLISHED: July 16, 1996 (19960716)
INVENTOR(s): YUBEERU KADEIMA
AREBU EDAN
APPLICANT(s): BULL SA [198400] (A Non-Japanese Company or Corporation), FR (France)
APPL. NO.: 07-196738 [JP 95196738]

FILED: August 01, 1995 (19950801)
PRIORITY: 94402197 [EP 94402197], EP (European Patent Office), October 03, 1994 (19941003)
INTL CLASS: [6] G06F-015/16; G06F-009/06; G06F-011/28
JAPIO CLASS: 45.4 (INFORMATION PROCESSING -- Computer Applications); 45.1 (INFORMATION PROCESSING -- Arithmetic Sequence Units)

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号

特開平8－185378

(43)公開日　平成８年(1996)７月16日

(51)Int.Cl.[6]		識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ	15/16	４５０　Ｚ			
	9/06	５４０　Ｓ			
	11/28		Ａ　7313－5Ｂ		

審査請求　有　　請求項の数５　ＯＬ　（全 6 頁）

(21)出願番号　特願平7－196738

(22)出願日　平成７年(1995)８月１日

(31)優先権主張番号　９４４０２１９７．１

(32)優先日　1994年10月３日

(33)優先権主張国　フランス（ＦＲ）

(71)出願人　390035633
ブル・エス・アー
フランス国、エフ－92800・ピュトー、プラス・カルポー、１、トウール・ブル（番地なし）

(72)発明者　ユベール・カデイマ
フランス国、78570・アンドルシー、リユ・ペンテイーブル・１

(72)発明者　アレブ・エダン
フランス国、78990・エランクール、シユマン・デ・ビーヌ・11

(74)代理人　弁理士　川口　義雄　（外２名）

(54)【発明の名称】　ＤＣＥ型分散情報処理環境内に分散したアプリケーションのデバッグ支援ツール

(57)【要約】

【課題】　ＤＥＣ分散環境下でのアプリケーションデバッグ支援ツールを提供する。

【解決手段】　デバッグ支援ツール（ＤＡＴ）は、中央プラットフォーム（ＭＦＣ）上に設置された、分散システムの全体的な状態を監視するプロセス管理手段（ＣＮＴ）と、各プラットフォーム上で走る各アプリケーション（ＡＳ１、ＡＳ２）の処理のテストの局所的総称動作をすべてサポートする、各副プラットフォーム（ＨＣ１、ＨＣ２）上に設置されたデバッグプロセス実行手段（ＡＧ１１、ＡＧ１２、ＡＧ２１、ＡＧ２２、ＡＧ２３）と、各々が、同一の副プラットフォームに設置された種々のプロセス実行手段と管理手段（ＣＮＴ）との間のインタフェースの役割を果たし、各々が、デバッグ動作によって生じるプロセス実行手段全体に共通な動作をすべて局所的に検証する、ＤＡＥＭＯＮプロセス手段の集合体（ＤＭ１、ＤＭ２）とを含む。

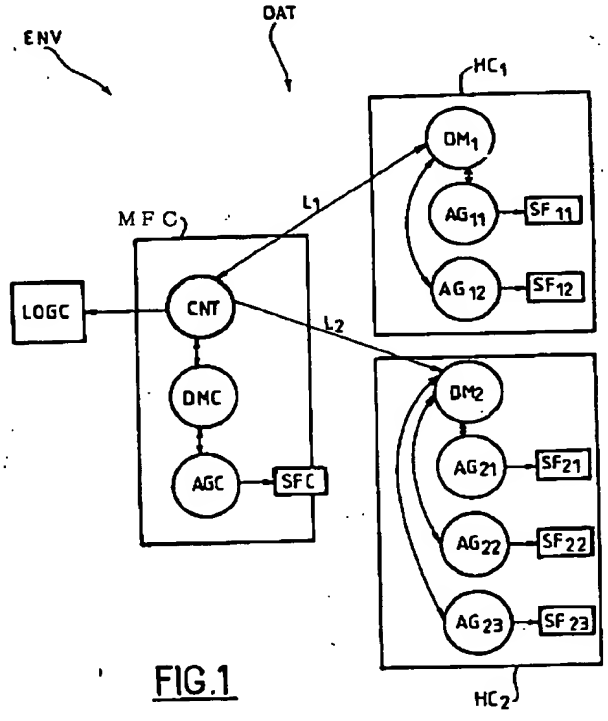

FIG.1

1

【特許請求の範囲】

【請求項１】　中央プラットフォーム（ＭＦＣ）上に設置された、分散システム全体の状態を監視するプロセス管理手段（ＣＮＴ）と、各プラットフォーム上で走る各アプリケーション（ＡＳ１、ＡＳ２）の処理のテストの局所的総称動作をすべてサポートしており、各副プラットフォーム（ＨＣ１、ＨＣ２）上に設置されたデバッグプロセス実行手段（ＡＧ１１、ＡＧ１２、ＡＧ２１、ＡＧ２２、ＡＧ２３）と、

各々が、同一の副プラットフォームに設置された種々のプロセス実行手段とプロセス管理手段（ＣＮＴ）との間のインタフェースの役割を果たし、各々が、デバッグ動作中のプロセス実行手段全体に共通な動作をすべて局所的に検証する、ＤＡＥＭＯＮプロセス手段の集合体（ＤＭ１、ＤＭ２）とを含んでおり、異なるオペレーティング・システムを有し、ネットワークによって相互接続されている複数の異種情報処理システム（ＭＦＣ、ＨＣ１、ＨＣ２）を含むＤＣＥ型の分散情報処理環境（ＥＮＶ）であって、アプリケーションが分散されている複数の副プラットフォーム（ＨＣ１、ＨＣ２）に結合されている中央プラットフォーム（ＭＦＣ）を含みその結果分散システムを形成する環境において、各アプリケーション（ＡＣ、ＡＳ１、ＡＳ２）を構成する全処理の試験を行い得ることを特徴とする分散アプリケーション（ＡＣ、ＡＳ１－ＡＳ２）のデバッグ支援ツール（ＤＡＴ）。

【請求項２】　プロセス実行手段（ＳＦＣ、ＳＦ１１、ＳＦ１２、ＳＦ２１、ＳＦ２２、ＳＦ２３）の１つにそれぞれ結合したファイルシステムであって、各ファイルが、各プラットフォーム上で、デバッグ中のすべての処理について、該処理のすべての大域状態を規定する、該処理に関するメッセージおよびイベントの記録簿を含むファイルシステムと、

－　デバッグ中の分散アプリケーションのすべての処理の大域状態が集中され、その結果、該アプリケーションが走る情報処理環境（ＥＮＶ）の大域状態を形成する、中央ファイル（ＬＯＧＣ）と、を含むことを特徴とする、請求項１に記載のツール。

【請求項３】　中央プラットフォーム（ＰＦＣ）が、デバッグ中のプログラムの実行の管理手段を含み、これら作業の管理が進化グラフィックインタフェースから行われることを特徴とする、請求項２に記載のツール。

【請求項４】　中央プラットフォーム（ＰＦＣ）が、ＣＨＡＮＤＹ、ＭＩＳＲＡ、ＨＡＡＳアルゴリズムから構成される、デバッグ中のアプリケーションプロセス間のデッドロック検出手段を含むことを特徴とする、請求項２に記載のツール。

【請求項５】　中央プラットフォームが、ＲＡＮＡアルゴリズムから構成されるプロセス終了検出手段を含むことを特徴とする、請求項２に記載のツール。

2

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、Open Software Foundation Inc社の１９９０年５月１４日付けの最新の刊行物「OSF Distributed Computing Environment Rationale」および１９９２年１月付けの「Distributed Computing Environment, an overview」に記載されているような、ＤＣＥ型分散情報処理環境内に分散したアプリケーションのデバッグ支援ツールに関する。

【０００２】

【従来の技術】このような環境は、たとえば異なる製造者によって製造され異なるオペレーティング・システムをもち、ネットワークによって相互接続されている複数の異種情報処理システムを含む。換言すれば、この環境は、走らせるアプリケーションが分散されているネットワークにおいて相互接続されている複数のプラットフォームまたはマシンから成る。上記環境はこのようにして分散システムを形成している。

【０００３】また、上記環境は、アプリケーションとネットワークに固有のソフトウエアの間に位置し、ネットワーク内に分散している異種のマシンの分散アプリケーションの開発、実行および運用を可能にするサービスの集合体を形成しているとも言うことができる。このようにして上記環境は、多数の製造者ネットワークにおけるデータおよびアプリケーションを分散し同期化する可能性をもたらす。

【０００４】

【発明が解決しようとする課題】数多くのテストと逐次分散プログラムとを必要とする、ＤＣＥ分散環境（またはＤＣＥ下分散ともいう）アプリケーションのデバッグ支援は困難できわめて複雑な問題であり、分散環境における並列処理を含むプログラムの場合なおさら困難できわめて複雑である。これは主に、同時実行（並列実行）されるものが決まらないため、実行時間中の挙動を再現するとが難しいことによる。

【０００５】市販されている逐次プログラムのデバッガの機能は、分散システムに固有の特徴を考慮するには充分でなく、また、ＤＣＥ上に分散したアプリケーションの開発支援のための産業用ツールは存在しない。

【０００６】したがって、高性能で有効なツールが使用可能であることが必須である。またこれらツールは、使い方が平易なプログラミング環境や、たとえばＸｗｉｎｄｏｗ／Ｍｏｔｉｆ形式のような進化したグラフィックインタフェースに挿入できるものでなければならない。

【０００７】本発明はこれらの要請に応えるもので、ＤＣＥ環境に分散したアプリケーションのデバッグ支援ツールに関する。該ツールにより、一方ではデバッグ作業に関わるすべてのプラットフォームと、他方ではそこで実行される処理の全体を見渡す特別なデバッグマシン（中央プラットフォーム、テストフォーカスマシンとも

呼ばれる）から、ある時点において実行中の全ての処理の状態を検証することが可能となる。

【０００８】

【課題を解決するための手段】本発明によれば、異なるオペレーティング・システムをもち、ネットワークによって相互接続されている複数の異種情報処理システムを含むＤＣＥ形の分散情報処理環境（ＥＮＶ）であって、アプリケーションが分散されている複数の副プラットフォームに結合されている中央プラットフォームを含みその結果分散システムを形成する環境における、分散アプリケーションのデバッグ支援ツールであって、各アプリケーションを構成する全ての処理の試験を行うことが可能なツールは、

－　中央プラットフォーム上に設置された、分散システムの全体的な状態を監視するプロセス管理手段と、

－　各プラットフォーム上で走る各アプリケーションの処理のテストの局所的総称動作をすべてサポートしており、各副プラットフォーム上に設置されたデバッグプロセス実行手段と、

－　各々が、同一のプラットフォームに設置された種々のプロセス実行手段とプロセス管理手段との間のインタフェースの役割を果たし、各々が、デバッグ動作によって生じるプロセス実行手段全体に共通な動作をすべて局所的に検証する、ＤＡＥＭＯＮプロセス手段の集合体と、を含むことを特徴とする。

【０００９】このようなツールによりＤＣＥ上のアプリケーションのデバッグに必要な２種類のテスト、すなわち開発テストと認可テスト、を実行することができる。前者は開発者の視点で実施されるもので、ユニットテスト、綜合テスト、全システムテストを含む。後者はユーザの視点によるもので、検収とメンテナンスという２つの目的で実施される。

【００１０】本発明の別の特徴および利点は、添付の図面を参照しつつ非限定的例として示した以下の説明を読むことにより明らかになろう。

【００１１】

【発明の実施の形態】図１について説明する。本発明によるアプリケーションデバッグ支援ツールＤＡＴは、ＤＣＥ形分散異種情報処理環境ＥＮＶ内に挿入される。該環境は以下の構成要素を含む。

【００１２】－　ＰＦＣなどの中央プラットフォーム。テストフォーカスマシンとも呼ばれる。

【００１３】－　副プラットフォーム。オブジェクトホストとも呼ばれる。図２では簡略化の目的から、たとえばＨＣ１とＨＣ２などのように２例しか例示しなかったが、副プラットフォームの数は任意の数とすることができることは言うまでもない。以後の文章中では、これら副プラットフォームは１つ（または複数）のテスト・セッションに参加しているマシンであるとする。また、これらマシンはテスト作業に参加しているマシンであると

言うこともできる。

【００１４】ＰＦＣ、ＨＣ１、ＨＣ２は、任意の種類（ＥＴＨＥＲＮＥＴ、ＦＤＤ１など）の同一のネットワークに接続される。ただ、図１の簡略化のためにネットワークは図示しない。ツールＤＡＴは、以下の主要な特徴的要素を含む。

【００１５】－　中央ホストＰＦＣ上に設置された、分散システムの全体的な状態を監視するプロセス管理手段ＣＮＴ、

－　アプリケーションデバッグプロセス実行手段。例として以下のものがある。

【００１６】－　ＰＦＣ上に設置されたＡＧＣ、

－　ＨＣ１上に設置されたＡＧ１１、ＡＧ１２、

－　ＨＣ２上に設置されたＡＧ２１、ＡＧ２２、ＡＧ２３、

したがってこれらは、環境ＥＮＶに属し各プラットフォーム上で走る各アプリケーションの処理のデバッグの局所的総称動作をすべてサポートする各プラットフォームに設置される。これらは、ＤＣＥ型環境でよく使われているたとえばＵＮＩＸオペレーティングシステム上で作動するあらゆるデバッガなど、あらゆる市販のデバッグソフトウェアと結合して作動する。

【００１７】－　ＤＡＥＭＯＮ処理。例として以下のものがある。

【００１８】－　ＭＦＣ上に設置されたＤＭＣ

－　ＨＣ１上に設置されたＤＭ１

－　ＨＣ２上に設置されたＤＭ２

したがってこれらも各ＥＮＶシステムに設置され、各々が、同一のサーバ上に設置された種々のプロセス実行手段（ＡＧＣに対するＤＭＣも同様である）とプロセス検証手段ＣＮＴとの間のインタフェースの役割を果たし、それらの間でそれぞれデバッグ動作によって生じるプロセス実行手段全体に共通な動作をすべて局所的に検証する。たとえばＵＮＩＸでは、ＤＡＥＭＯＮ処理は、スリーピング処理であってイベント発生時起動する処理であると定義されている。イベントとは、分散アプリケーションすべてについてデバッグが必要となった場合、ユーザがＰＦＣから起動するコマンドであると定義される。

【００１９】－　たとえば、それぞれＳＦＣがＡＣに、ＳＦ１１、ＳＦ１２がＡＧ１１、ＡＧ１２に、ＳＦ２１、ＳＦ２２、ＳＦ２３がそれぞれＡＧ２１、ＡＧ２２、ＡＧ２３に結合しているというように、各々が各プロセス実行手段に結合しているファイルシステム。また中央ファイルＬＯＧＣは管理手段ＣＮＴに結合している。

【００２０】以下の文中においては、プロセス管理手段、プロセス実行手段、ＤＡＥＭＯＮプロセス手段を単に各々、管理、実行、ＤＡＥＭＯＮと呼ぶことにする。

【００２１】ツールＤＡＴの機能上の主な特徴は以下の通りである。

【００２２】－　１）たとえばＰＦＣなどの唯１つの中央デバッグマシンを基にしてＥＮＶ内で実行される処理の全体的な状態をチェックすること。

【００２３】－　２）管理ＣＮＴと作用体と、デバッグ作業にかかわる種々のプラットフォーム、ここではＨＣ１とＨＣ２、に設置されたＤＡＥＭＯＮとの間の交信ベクトルとしてＤＣＥ内で使用される、ＰＲＣ形式の既知の交信手順を使用すること。この観点から見れば、テストＤＭ１２、ＤＭ２などの種々のＤＡＥＭＯＮに対し、ＣＮＴはクライアントＤＣＥであり、これらテストＤＭ１２、ＤＭ２などの種々のＤＡＥＭＯＮは、ＣＮＴに対してはサーバＤＣＥであり、作用体ＡＧ１１－ＡＧ１２、ＡＧ２１－ＡＧ２２－ＡＧ２３に対しては各々クライアントＤＣＥであり、また作用体自身もサーバである。もちろんクライアントとサーバという概念はＤＣＥに関する前記の文献で定義されている。

【００２４】－　３）デバッグ中に発生したイベントを検証し（トレースを実行するとも言う）、デバッグ・セッションに関わる各プラットフォーム－ここではＰＦＣ、ＨＣ１、ＨＣ２－で発生したイベントであって、したがってテストされる各アプリケーションに影響を及ぼすイベント間の原因の依存の特徴を究明するためのベクトルクロック機構を使用することにより、その原因に関する相互依存性の調査を実施すること。

【００２５】－　４）デバッグ中のシステムの作動の一貫性を維持すること。

【００２６】－　５）処理とスレッド間のデッドブロックを検出すること。

【００２７】－　６）ＰＦＣ、ＨＣ１、ＨＣ２などにおける、あるアプリケーションの全処理が終了したことを検出すること。

【００２８】上記特徴の１に関しては、マシンＰＦＣは、デバッグ・セッションにかかわる別のすべてのマシンの集合体（ここではＨＣ１とＨＣ２）を見渡す。

【００２９】デバッグ中のプログラムの実行の検証作業はすべてこのレベルで行われ、対応するすべての作業の管理は、たとえばＸ／Ｗｉｎｄｏｗ／Ｍｏｔｉｆ形式の進化グラフィックインタフェースを基点にして行われる。なお、デバッグ・セッション時には、ある時点においてすべてのマシンが終了していなければならない。また、以下の作業が行われるのもＰＦＣのレベルである。

【００３０】－　処理間のデッドブロックを検出すること

－　処理の終了を検出すること

－　デバッグ・セッションにかかわるすべてのマシンに関し、テストされる各アプリケーションの種々の処理のデバッグ履歴に関するすべてのデータを集約すること。テストをまとめた結果を活用することにより、あるアプリケーションの処理デバッグに影響を及ぼす種々のイベント間の原因に関しどちらが先に発生したかという関係

が明らかになる。当業者は通常、デバッグ履歴に関するデータを示すのにＬｏｇという用語を用いているので、以後の本文ではその用語を使用することにする。データはＬｏｇファイルと呼ばれるファイルに格納される。同ファイルは各作用体に結合されたファイル、すなわちＡＧ１１についてはＳＦ１１、ＡＧ１２についてはＳＦ１２、に挿入される。

【００３１】ＣＮＴによるプログラムの実行の検証は、全体すなわちＥＮＶ内におけるＤＣＥシステムの全体的な状態の検証によって行われる。これにより、

－　ａ）時間経過におけるプログラムの実行の等価性（プログラムは時間経過中、同一の時点で実行されなければならない）と、

－　ｂ）エラーが発生した前後関係でプログラムを再起動できるようなエラーが検出された場合の、プログラムの実行再現性と、

－　ｃ）デバッグ時に発生するイベントおよびメッセージの日報処理の際のＤＣＥシステムの作動の一貫性と、

が確保される。

【００３２】日報処理とはＬｏｇファイルにこれらメッセージおよびイベントのすべての履歴を記録することである。ある分散プログラムの実行の再現性は、それが不確定的であることとメッセージの伝送時間がまちまちであることにより実現は困難である。そのような理由から、実行中の処理の回復可能な全体的な状態を効果的に検証できるようにならなければならない。

【００３３】ある全体的な状態は、それが一貫性をもち最後のセーブポイント以降種々の処理が受信したメッセージを含む時、回復可能であると言う。そのような状態は、実行中のすべての処理の状態とこれら処理間にある通信チャンネルとから成る集合によって構成される。分散環境における難しさの主たるものは、異種機で実行される種々の処理の全体的な状態の回復が可能であるかどうかである。

【００３４】本発明によるツールによってこの問題にもたらされた解決方法は、処理間で交換されたメッセージの日報処理と、各処理の状態の日報処理である。ある時点において全体的な状態を日報処理することはほとんど不可能であるため、後者の日報処理は各処理毎に独立してしか行うことができない。

【００３５】各処理について、それに結合されているＬｏｇファイルにおいて、受信メッセージおよびその状態のセーブポイントが日報処理される。

【００３６】その場合のチェッカＣＮＴの役割は、テスト作業に関わる種々のマシンＨＣ１、ＨＣ２．．．などの処理の日報を定期的に収集することであり、したがって、このレベルにおいては日報はＬＯＧＣファイルに集められる。

【００３７】これを行うため、たとえば、１９９０年付「journal of algorithms」誌、４６２～４９１ページ

のD．B．ＪｏｈｎｓｏｎおよびW．Ｓｗａｎｅｐｏｅｌの資料「Recovery in distributed systems using optimistic message logging and checkpointing」において説明されている技術を使用する。この技術の骨子は、全体的な同期を行わずに処理間で交換されたメッセージのすべてと、メッセージ受信時の処理の状態とを日報処理することである。回復可能最終状態の計算が行われるのは、停止後の再起動中である。

【００３８】この計算は、たとえば、１９９０年付「10th international conference on distributed computing systems」誌の１２～１９ページ所載のＭ．Ａｈｕｊａ、Ａ．Ｄ．Ｋｓｈｅｍｋａｌｙａｎｉ、Ｔ．Ｃａｒｌｓｏｎの資料『A basic unitof computation in distributed systems』において説明されているような分散アルゴリズムにより行うことができる。

【００３９】図２を参照すると、ホストＭＦＣ上で走るクライアント・アプリケーションＡＣと、ＨＣ１およびＨＣ２上で走る２つのサーバ・アプリケーションＡＳ１およびＡＳ２との間の分散アプリケーションの場合の、ＥＮＶの構成要素間の種々の相互作用が示されていることがわかる。前記のようにＡＣは、ＲＰＣ形式の接続線Ｌ１、Ｌ２により各々、ＡＳ１およびＡＳ２と交信する。

【００４０】同図にはさらに、管理ＣＮＴと、たとえばＳＭＰＣ、ＳＭＰ１、ＳＭＰ２などの、ＡＣ、ＡＳ１、ＡＳ２に各々対応するデバッグセッション間の接続線を示した。これらの接続線もＲＰＣ形式であり、各々Ｌ、Ｌ’１、Ｌ’２で示されている。これら接続線を通して、ＣＮＴは、ＨＣ１、ＨＣ２．．．などの種々のマシン上で実施される処理の全体的な状態の日報を収集する。

【００４１】ある分散プログラムのデバッグにあたっては、該プログラムの相次ぐ状態と、その構成要素のある状態から別の状態への移行のトリガを行う全ての情報を検証し、原因に関する相互依存性を明らかにするためこれらイベント間に順序関係を導入することが必要である。

【００４２】テスト中に発生したイベントの検証およびその原因に関する相互依存性の調査は、すべてのマシン上で実施されたあらゆる処理のイベント間の原因の依存性の特徴を究明するためのベクトルクロック機構をＥＮＶ内で使用することにより実施する。この機構については、たとえば、１９８８年付のＮｏｒｔｈ Ｈｏｌａｎｄ社発行の「international conference on parallel and distributed algorithms」誌２１５～２２６ページに所載のＦ．Ｍａｔｔｅｒｎの「Virtual time and global states in distributed systems」と題する論文に説明がある。

【００４３】デバッグ中のシステムの一貫性の確保にあたっては、システムの全体的な状態を反映した信頼のおける日報の更新が必要である。これは、たとえば、１９８５年８月付「ACM transactions and computer systems」誌、第３巻第３号、２０４～２２６ページに所載のＲ．Ｅ．ＳｔｒｏｍおよびＳ．Ｙｅｍｉｎｉの論文「Optimistic recovery in distributed systems」において示されているように、管理ＣＮＴ側にメッセージの日報を生成することにより達成される。デバッグ中、ユーザは種々の処理間のデッドロックの検出を起動することができる。

【００４４】デッドロックとは、ある処理の集合体の各要素が、同一の集合体に属する別の処理によってしか発生し得ないイベントの発生を待っている状態である。この状態は待機グラフによって表すことができる。分散環境下では、処理の集合体Ｓのデッドロック条件は、Ｓのある要素についての次要素がすべてＳ内にあるということである。ＥＮＶなどのＤＣＥ環境下では、デッドロックは通信を原因とする。本発明によれば、このデッドロックの問題を、１９８３年５月付「ACM TOCS」誌、第１巻第２号、１４４～１５６ページに記載の「distributed deadlock detection」と題する論文に記載されている、Ｃｈａｎｄｙ、Ｍｉｓｒａ、Ｈａａｓのアルゴリズムで解決する。

【００４５】システムがテスト実行中に停止した（条件付きブレークポイントに達し、ユーザ（分散プログラムのデバガ）により例外または停止が発生した）場合、テスト実行中のシステムの一貫性の維持の問題が生じる。実際、処理全体が瞬時に停止するのではなく、実行のグラニュラリティ（プログラミングにおいては、実行のグラニュラリティはあるプログラム内の実行シーケンスの長さである）に必ずしも到達するとは限らない。その理由は、すべての処理が受動的状態（すなわち活動していない状態）であるとみなされてもそれでアプリケーションが終了したということを示している訳ではないからである。実際には、また観察されていず次に受動的状態になる処理によって発せられたメッセージによって、受動的状態であるとみなされた処理が再度起動されることがある。

【００４６】Ｒａｒａアルゴリズムにより、ユーザからの要求があった場合ＤＣＥ環境内での処理の実行の終了を検出することができる。同アルゴリズムは、１９８３年７月１７日付「Information processing letter」誌、第１７巻、４３～４６ページに所載のＳ．Ｐ．Ｒａｎａの論文「A distributed solution of the distributed termination problem」に記載されている。

【図面の簡単な説明】

【図１】本発明による、アプリケーション開発支援ツールの全体的アーキテクチャを示す図である。

【図２】本発明による、ツール内で行われるＤＣＥ上クライアント・サーバアプリケーションのテストの種々の相互作用を示す図である。

【図１】

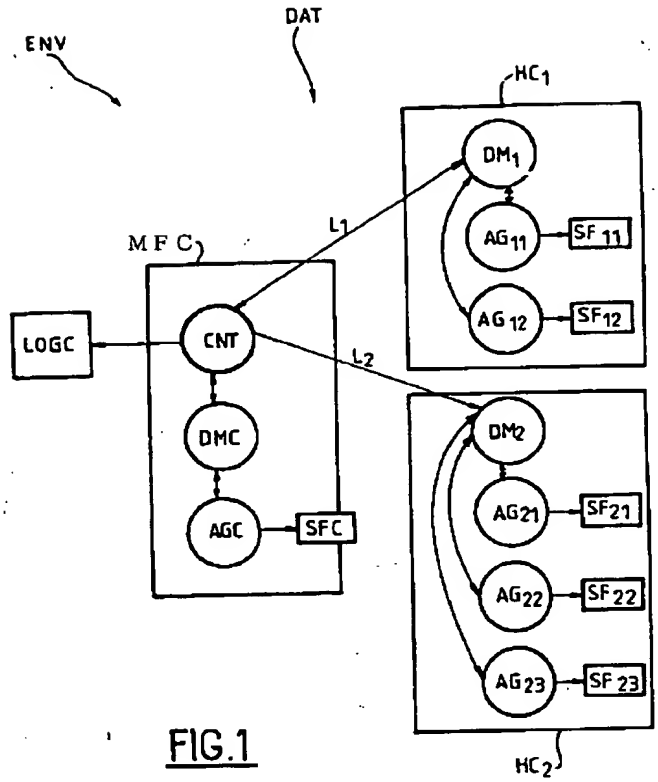

FIG.1

【図２】

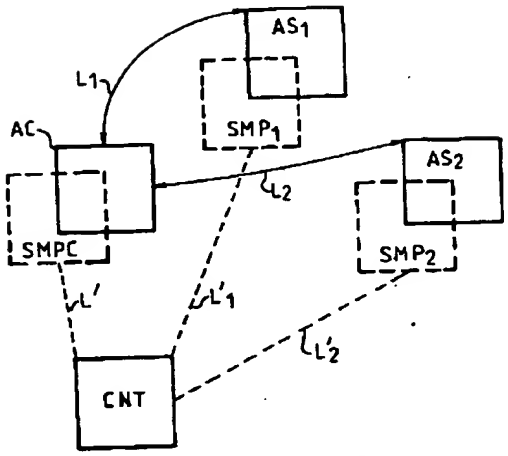

FIG.2